UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/796,358	03/08/2004	Matthew Conover	SYMC1057	2266

34350 7590 05/31/2007
GUNNISON, MCKAY & HODGSON, L.L.P.
1900 GARDEN ROAD, SUITE 220
MONTEREY, CA 93940

EXAMINER	
AYASH, MARWAN	
ART UNIT	**PAPER NUMBER**
2185	

MAIL DATE	DELIVERY MODE
05/31/2007	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Office Action Summary	Application No.	Applicant(s)
	10/796,358	CONOVER, MATTHEW
	Examiner	Art Unit
	Marwan Ayash	2185

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 March 2004*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-25* is/are pending in the application.

4a) Of the above claim(s) *21-24* is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-20 and 25* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on *08 March 2004* is/are: a)☒ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)	4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)	5) ☐ Notice of Informal Patent Application
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date *5/27/04*.	6) ☐ Other: _____.

DETAILED ACTION

Election/Restrictions

1. Restriction to one of the following inventions is required under 35 U.S.C. 121:

   I. Claims 1-20, 25 drawn to stalling allocation/deallocation requests to determine where specific block pointers point, classified in class 711, subclass 170.

   II. Claims 21-24, drawn to detecting a corrupt linked list and repairing or reinitializing the list, classified in class 714, subclass 27.

2. The inventions are distinct, each from the other because of the following reasons:

3. Inventions I and II are related as subcombinations disclosed as usable together in a single combination. The subcombinations are distinct if they do not overlap in scope and are not obvious variants, and if it is shown that at least one subcombination is separately usable. In the instant case, subcombination II has separate utility such as being usable in any computer system in which a linked list is implemented as a data structure to keep track of any information/data. This subcombination is not required to be used along with subcombination I. See MPEP § 806.05(d).

Because these inventions are distinct for the reasons given above and a different search is required for each invention, restriction for examination purposes as indicated is proper. Furthermore, because these inventions are distinct for the reasons given above and have acquired a separate status in the art as shown by their different classification, restriction for examination purposes as indicated is proper.

The examiner has required restriction between combination and subcombination inventions. Where applicant elects a subcombination, and claims thereto are subsequently found allowable, any claim(s) depending from or otherwise requiring all the limitations of the allowable subcombination will be examined for patentability in accordance with 37 CFR 1.104. See MPEP § 821.04(a). Applicant is advised that if any claim presented in a continuation or divisional application is anticipated by, or includes all the limitations of, a claim that is allowable in the present application, such claim may be subject to provisional statutory and/or nonstatutory double patenting rejections over the claims of the instant application

A telephonic provisional election was made without traverse to prosecute the claims of invention I by Serge Hodgson on 5/15/07. Affirmation of this election must be made by applicant in replying to this office action. Claims 21-24 are withdrawn from further consideration by the examiner, 37 CFR 1.142(b), as being drawn to a non-elected invention.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

Claim 25 is rejected under 35 U.S.C. 101 because the claimed invention is directed to non-statutory subject matter. The claim is directed to a computer program product comprising a computer-readable medium containing computer program code. Applicant's disclosure [*pages 32 last paragraph – first incomplete paragraph of page 33*] provides intrinsic evidence that would lead one of ordinary skill to reasonably interpret the claimed computer program product as a signal transmitted over a network. As such, one of ordinary skill would reasonably interpret the computer program product as the transmission itself, which is a form of energy and not manufactures, machines, processes or compositions of matter and is not believed to be functionally and structurally interconnected with the program/code in the signal in such a manner as to enable it to act as a computer component and realize it's functionality. Thus no usefulness of the program/code may be realized from the claimed computer program product.

Claim Rejections - 35 USC § 112

5. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

6. **Claim 14 is rejected** under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention because it is unclear what is meant by "determining said block" such that the scope of the claims is indefinite.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the

subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

8. The factual inquiries set forth in *Graham* v. *John Deere Co.*, 383 U.S. 1, 148 USPQ 459 (1966), that are applied for establishing a background for determining obviousness under 35 U.S.C. 103(a) are summarized as follows:

1. Determining the scope and contents of the prior art.
2. Ascertaining the differences between the prior art and the claims at issue.
3. Resolving the level of ordinary skill in the pertinent art.
4. Considering objective evidence present in the application indicating obviousness or nonobviousness.

9. **Claims 1-20, 25 are rejected** under 35 U.S.C. 103(a) as being unpatentable over Fetzer (US Patent # 6,832,302) in view of Abrashkevich (US PGPub # 2004/0221120).

With respect to **independent claims 1, 25** Fetzer discloses: A method/computer program product [*Fetzer – claim 11*] comprising:

Stalling [*wrapping and/or intercepting*] a heap allocation function call to a heap allocation function originating from a request by an application for a block of heap buffer [*wrapping allocation functions (such as malloc, calloc, free, etc.) (Fetzer – Col 2 lines 26-38, Col 3 lines 42-61, Fig. 3 steps 301-302)*];

predicting a predicted block of said heap buffer to fulfill said request [*Fetzer – Col 9 line 58 – Col 10 line 15, Col 10 lines 30-45*];

Fetzer does not explicitly disclose: Determining if a forward link (F-link) and a backward link (B-link) of said predicted block are addresses within a heap segment associated with said predicted block, although Fetzer suggests this limitation by disclosing that the buffer space may be kept track of by wrapping allocation functions (such as malloc, calloc, free, etc.) and keeping meta-data for each allocated buffer (*Fetzer – Col 2 lines 34-38*). Furthermore Fetzer discloses that strcpy fault-containment wrapper 131 may determine if destination start address 163 is part of any allocated buffer in heap 160. If the destination start address does not fall within a heap buffer, then an error handling procedure is executed (*Fetzer – Col 6 lines 34-37*). It is noted that Fetzer implements a buffer management table as a special data structure for maintaining header information and other meta-data about each buffer or allocated block in the heap (*Fetzer – Col 7 lines 50-66*), but also mentions that other data structures may be used, instead of the buffer management table, such as linked lists to keep track of allocated/free blocks of memory in the heap (as in the instant application) (*Fetzer – Col 8 lines 16-18*).

In the same field of endeavor, Abrashkevich teaches defensive heap memory management (*Abrashkevich – title*) wherein doubly linked lists are implemented (including an f-link and a b-link) as memory management mechanisms to manage and keep track of allocated and free blocks/chunks of memory (*Abrashkevich – paragraphs 0003, 0025, first ~17 lines of paragraph 0026, 0029*).

Therefore, Fetzer in view of Abrashkevich discloses all limitations of the instant claim(s) including:

predicting a predicted block of said heap buffer to fulfill said request [*Abrashkevich – last 10 lines of the left-hand column of paragraph 0025, and paragraph 0032*];

determining if a forward link (F-link) and a backward link (B-link) of said predicted block are addresses within a heap segment associated with said predicted block [*allocated chunks are stored in a doubly linked skip list (each chunk has an f-link and b-link associated with it) sorted by their offsets and free chunks are stored in a doubly linked skip list sorted by their offsets and sizes... each node of both skip lists contains, inter alia, chunk offset (offset to a memory pool/block from the beginning of a heap that is returned to a process, which has requested an allocation, as an address or as a part of a handle (Abrashkevich – paragraph 0029). Thus knowing this address/offset, a determination may be made with respect to if an f-link and b-link (the chunks before and after an allocated chunk in an ordered/sorted linked list) are addresses within a heap segment associated with the allocated chunk. Furthermore, Abrashkevich discloses checking allocated memory blocks for leaks and other errors (Abrashkevich – paragraph 0026), and freeing corrupted allocation by checking a block's/chunk's left and right neighbors for corruption (Abrashkevich – middle of paragraph 0048, and described in more detail in paragraph 0069). Lastly, see paragraph 0035 of Abrashkevich wherein block/chunk deallocation is described, involving checking a block's/chunk's f-link and b-link to determine the possibility of coalescing memory blocks/chunks if they have not been corrupted*].

It would have been obvious to one having ordinary skill in the art at the time the invention was made to determine if a forward link (F-link) and a backward link (B-link) of an allocated or free block are addresses within a heap segment associated with the predicted block in the invention of Fetzer as taught by Abrashkevich because such an implementation would allow for verification of memory corruption (*Abrashkevich – paragraph 0069*) consistent with the inventive concepts of defensive heap memory management, and means for detecting heap smashing.

With respect to **independent claim 11** Fetzer discloses: A method comprising:

stalling [*wrapping and/or intercepting*] a heap deallocation function call to a heap deallocation function originating from a release by an application of a block of heap buffer [*wrapping allocation/deallocation functions (such as malloc, calloc, free, etc.) (Fetzer – Col 2 lines 26-38, Col 3 lines 42-61, Fig. 3 steps 301-302)*],

Fetzer does not explicitly disclose:

wherein said block is a deallocation block that is being deallocated to a deallocation freelist [*although Fetzer suggests this limitation by disclosing a buffer management table as a special data structure for maintaining header information and other meta-data about each buffer or allocated block in the heap (Fetzer – Col 7 lines 50-66), in addition to mentioning that other data structures may be used, instead of the buffer management table, such as linked lists to keep track of allocated/free blocks of memory in the heap (as in the instant application) (Fetzer – Col 8 lines 16-18)*];

determining if a forward link (F-link) of a list head of said deallocation freelist and a backward link (B-link) of a first block of said deallocation freelist are addresses within a heap segment associated with said deallocation freelist [*although Fetzer suggests this limitation by disclosing that the buffer space may be kept track of by wrapping allocation functions (such as malloc, calloc, free, etc.) and keeping meta-data for each allocated buffer (Fetzer – Col 2 lines 34-38). Furthermore Fetzer discloses that strcpy fault-containment wrapper 131 may determine if destination start address 163 is part of any allocated buffer in heap 160. If the destination start address does not fall within a heap buffer, then an error handling procedure is executed (Fetzer – Col 6 lines 34-37)*].

In the same field of endeavor, Abrashkevich teaches defensive heap memory management (*Abrashkevich – title*) wherein doubly linked lists are implemented (including an f-link and a b-link) as memory management mechanisms to manage and keep track of allocated and free blocks/chunks of memory (*Abrashkevich – paragraphs 0003, 0025, first ~17 lines of paragraph 0026, 0029*).

Therefore, Fetzer in view of Abrashkevich discloses all limitations of the instant claim(s) including:

wherein said block is a deallocation block that is being deallocated to a deallocation freelist [*Abrashkevich – paragraph 0031, 0035*];

determining if a forward link (F-link) of a list head of said deallocation freelist and a backward link (B-link) of a first block of said deallocation freelist are addresses within a heap segment associated with said deallocation freelist [*free (deallocated) chunks are stored in a doubly linked skip list (each chunk has an f-link and b-link associated with it) sorted by* *their offsets and sizes... each node of both skip lists contains, inter alia, chunk offset (offset to a memory pool/block from the beginning of a heap that is returned to a process as an address or as a part of a handle (Abrashkevich – paragraph 0029). Thus knowing this address/offset, a determination may be made with respect to if an f-link and b-link (the chunks before and after a deallocated chunk in an ordered/sorted linked list) are addresses within a heap segment associated with the deallocated chunk. Furthermore, Abrashkevich discloses checking allocated memory blocks for leaks and other errors (Abrashkevich – paragraph 0026), and freeing corrupted allocation by checking a block's/chunk's left and right neighbors for corruption (Abrashkevich – middle of paragraph 0048, and described in more detail in paragraph 0069). Lastly, see paragraph 0035 of Abrashkevich wherein block/chunk deallocation is described, involving checking a block's/chunk's f-link and b-link to determine the possibility of coalescing memory blocks/chunks if they have not been corrupted, this entails determining where the f-link and b-link associated with a deallocated block point*].

It would have been obvious to one having ordinary skill in the art at the time the invention was made to determine if a forward link (F-link) of a list head of a deallocation freelist and a backward link (B-link) of a first block of a deallocation freelist are addresses within a heap segment associated with the deallocation freelist in the invention of Fetzer as taught by Abrashkevich because such an implementation would allow for verification of memory corruption (*Abrashkevich – paragraph 0069*) consistent with the inventive concepts of defensive heap memory management, and means for detecting heap smashing. In addition, such an implementation would provide means for determining the possibility of coalescing neighboring free blocks (*Abrashkevich – paragraph 0035*).

With respect to **dependent claims 2, 13** as applied to claims 1, 11 above Fetzer in view of Abrashkevich discloses hooking said heap allocation function [*When a program executes a request to allocate memory on the heap (for example a p=malloc(size) instruction), this call is intercepted and the allocation wrapper is invoked (Fetzer – Col 10 lines 30-45)*].

With respect to **dependent claim 3** as applied to claim 1 above Fetzer in view of Abrashkevich discloses determining a size of said block [*Buffer management table 410 is a separate table that stores the size and the address of each allocated buffer in the heap (Fetzer – Col 8 lines 3-8)*], [*Abrashkevich – paragraph 0003, 0025, 0029*].

With respect to **dependent claim 4** as applied to claim 3 above Fetzer in view of Abrashkevich discloses said predicted block has said size [*Abrashkevich – paragraph 0032*].

With respect to **dependent claim 5** as applied to claim 3 above Fetzer in view of Abrashkevich discloses a freelist comprises a plurality of free blocks having said size, said predicted block being on said freelist [*list of free chunks (Abrashkevich – paragraph 0031, 0032)*].

With respect to **dependent claim 6** as applied to claim 1 above Fetzer in view of Abrashkevich discloses said predicted block is on a predicted freelist [*Abrashkevich – paragraph 0032*].

With respect to **dependent claim 7** as applied to claim 6 above Fetzer in view of Abrashkevich discloses determining whether a F-link of a predicted list head of said predicted freelist points into said heap segment [*allocated chunks are stored in a doubly linked skip list (each chunk has an f-link and b-link associated with it) sorted by their offsets and free chunks are stored in a doubly linked skip list sorted by their offsets and sizes... each node of both skip lists contains, inter alia, chunk offset (offset to a memory pool/block from the beginning of a heap as an address or as a part of a handle (Abrashkevich – paragraph 0029). Thus knowing the address/offset, a determination may be made with respect to if an f-link or b-link (the chunks before and after an allocated/free chunk in an ordered/sorted linked list) are addresses that point into a heap segment associated with the allocated/free chunk. See also paragraph 0069 wherein examining next and previous chunks/blocks for memory corruption is discussed in detail*].

With respect to **dependent claim 8** as applied to claim 6 above Fetzer in view of Abrashkevich discloses determining whether a B-link of a predicted next block of said predicted freelist points into said heap segment [*allocated chunks are stored in a doubly linked skip list (each chunk has an f-link and b-link associated with it) sorted by their offsets and free chunks are stored in a doubly linked skip list sorted by their offsets and sizes... each node of both skip lists contains, inter alia, chunk offset (offset to a memory pool/block from the beginning of a heap as an address or as a part of a handle (Abrashkevich – paragraph 0029). Thus knowing the address/offset, a determination may be made with respect to if an f-link or b-link (the chunks before and after an allocated/free chunk in an ordered/sorted linked list) are addresses that point into a heap segment associated with the allocated/free chunk. See also paragraph 0069 wherein examining next and previous chunks/blocks for memory corruption is discussed in detail*].

With respect to **dependent claims 9, 16** as applied to claims 1, 11 above Fetzer in view of Abrashkevich discloses upon a determination that said F-link and said B-link of said predicted block are not addresses within said heap segment, said method further comprising taking corrective action [*if a memory chunk or one of it's neighbors is corrupt, corrective action* *such as freeing/deallocating the corrupt chunks may be performed depending on the error recovery protocol; also a pointer (likely altered during an attack or otherwise invalid) causes the corruption (Abrashkevich – paragraph 0069)*].

With respect to **dependent claim 10** as applied to claim 9 above Fetzer in view of Abrashkevich discloses taking corrective action comprises setting said F-link and said B-link to be an address of a list head of a freelist comprising said predicted block [*corrective action such as freeing/deallocating the corrupt chunks may be performed depending on the error recovery protocol (Abrashkevich – paragraph 0069), and since the free list may be a circular doubly linked list (Abrashkevich – paragraph 0031), both pointers associated with the free block may point to the list head assuming it is the only free block in the list*].

With respect to **dependent claim 12** as applied to claim 11 above Fetzer in view of Abrashkevich discloses reading said F-link and said B-link [*Abrashkevich – paragraph 0035*].

With respect to **dependent claim 14** as applied to claim 9 above Fetzer in view of Abrashkevich discloses determining said block [*a determination is made with respect to a free block (Abrashkevich – paragraph 0035)*].

With respect to **dependent claim 15** as applied to claim 11 above Fetzer in view of Abrashkevich discloses upon a determination that said F-link and said B-link are addresses within said heap segment [*check is performed following links in both directions, at which point a determination can be made that both links point to addresses within the heap segment associated with the free block (Abrashkevich – paragraph 0035)*; see *also claims 7 and 8 above since this limitation is substantially similar to the limitations presented in those claims*], said method further comprising releasing said heap deallocation function call [*if function call will not smash the heap, the request/call is released, or otherwise allowed to proceed (Fetzer – Fig. 3 step 309, Col 2 lines 50-58)*].

With respect to **dependent claim 17** as applied to claim 11 above Fetzer in view of Abrashkevich discloses if said F-link or said B-link is a stray F-link or stray B-link, said method further comprising determining if said stray F-link or stray B-link is a known false positive [*if a corrupt pointer is detected, means are provided using a memory debug information area associated with each chunk to facilitate checking memory for corruption (Abrashkevich – paragraph 0069)*].

With respect to **dependent claim 18** as applied to claim 11 above Fetzer in view of Abrashkevich discloses determining if said block is to be coalesced with other free blocks [*Abrashkevich – paragraph 0035*].

With respect to **dependent claim 19** as applied to claim 11 above Fetzer in view of Abrashkevich discloses wherein said block is to be coalesced with a coalesced block, said method further comprising: determining if a F-link and a B-link of said coalesced block are addresses within a heap segment associated with said coalesced block [*check is performed following links in both directions, at which point a determination can be made that both links point to addresses within the heap segment associated with the free block (Abrashkevich – paragraph 0035)*].

With respect to **dependent claim 20** as applied to claim 19 above Fetzer in view of Abrashkevich discloses determining if there are other blocks to be coalesced with said block [*Abrashkevich – paragraph 0004, 0031, 0035*].

Conclusion

The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. When responding to the office action, applicants are advised to clearly point out the patentable novelty which they think the claims present in view of the state of the art disclosed by the references cited or the objections made. Applicants must also show how the amendments avoid such references or objections. See 37 C.F.R. 1.111(c). In addition, applicants are advised to provide the examiner with the line numbers and page numbers in the application and/or references cited to assist examiner in locating the appropriate paragraphs.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Marwan Ayash whose telephone number is 571-270-1179. The examiner can normally be reached on Mon-Fri 9am-6pm.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Sanjiv Shah can be reached on (571)272-4098. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

Marwan Ayash
Examiner
Art Unit 2185

5/16/07

[illegible]

SANJIV SHAH
SUPERVISORY PATENT EXAMIN
TECHNOLOGY CENTER 21